Sheet-Based Hexahedral Mesh Generation

Jason F. Shepherd

Sandia National Laboratories
PO Box 5800 MS0376
Albuquerque, NM 87185-0376
jfsheph@sandia.gov

Historically, hexahedral finite element mesh generation has required significant geometric decomposition with the resulting consequence that generating hexahedral meshes can be extremely difficult to perform and automate and the process often takes several orders of magnitude longer in time to complete than current methods for generating tetrahedral meshes. In this work, we present methods for generating and modifying hexahedral meshes based on direct manipulation of the hexahedral sheets underlying a given hexahedral mesh. Using direct sheet-based techniques, we can generate hexahedral meshes in complex parts with significantly less decomposition of the original geometry over traditional methods.

1 Introduction

A hexahedral mesh is defined by a collection of layers of hexahedral elements. The layers of hexahedra can be arranged with a great deal of structure, or may be intersecting in extremely unstructured arrangements. Each of the layers of hexahedra can also be visualized in dual form as a manifold surface within the boundary of the space occupied by the mesh.

Because of the complexity often associated with hexahedral meshing, viewing a hexahedral mesh as a collection of surfaces, or sheets, can have many advantages in elucidating how the mesh is built and arranged. Methods for manipulating sheets (including adding or removing sheets) can be utilized to adapt, define features, and improve structure within these meshes. Additionally, defining required sheets within a geometry can enable the building of hexahedral meshes in complex geometries which have historically required significant geometric decomposition into sub-volumes which can be meshed using a known hexahedral template.

In this paper we demonstrate how to capture geometric boundary features using hexahedral sheets. For a more detailed description of the underlying theory utilized in this paper, we refer the reader to [5]. We will demonstrate

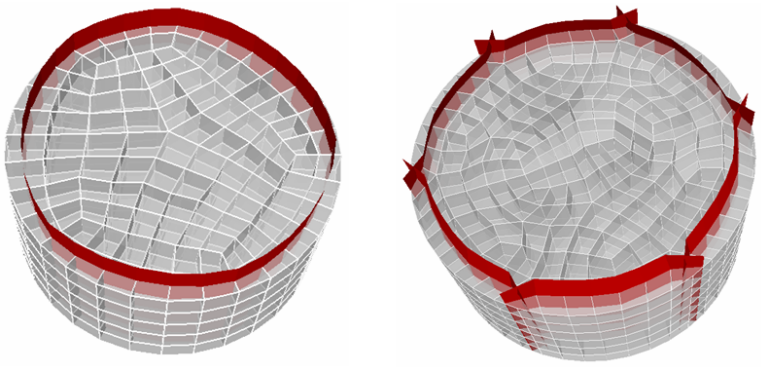

Fig. 1. Image showing how a sheet captures the geometric boundary. The picture on the right shows a single sheet capturing the cylindrical surface, while the picture on the left (of a different mesh) shows the same surface being captured with multiple sheets.

two examples where introduction of new sheets enables meshing of geometries which historically have been difficult to mesh or require significant geometric decomposition prior to obtaining a hexahedral mesh.

2 Methods

As discussed in [5], a minimal requirement for capturing a geometric surface in a hexahedral mesh is at least one of the hexahedral sheets collocated near the geometric surface within the mesh (see Figure 1). A similar requirement exists for geometric curves and chords in the mesh (shown in Figure 2). Strategic placement of new sheets can, therefore, enable new or additional geometric features to be added to an existing mesh, as shown in the examples later in this paper.

3 Examples

In this section, we will demonstrate two examples of inserting hexahedral sheets improve the geometric fidelity of a existing mesh to a given geometry. The two examples shown are provided courtesy of ANSYS [2], and the sheet modification operations were accomplished using SCIRun [1] and CUBIT [3].

3.1 Engine Model

Hexahedral mesh generation of the engine model is difficult due to the coupled geometry of the interior cavity with the actual engine block geometry (i.e., a

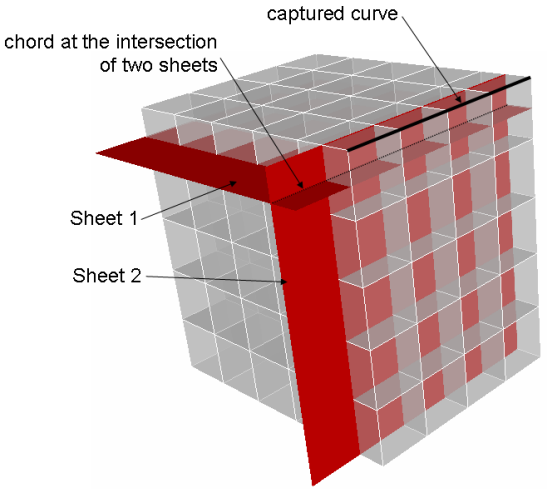

Fig. 2. Capturing a curve utilizing an offset chord (from the intersection of two sheets).

conformal mesh of both pieces is desired). Decomposition for meshing of both pieces separately may be accomplished resulting in primitive objects which readily accept a hexahedral mesh; however, the coupling of both pieces in the hexahedral mesh requires a significant amount of additional decomposition to generate a conformal mesh of both the engine block and the interior chamber of the engine.

The hexahedral mesh of the engine model, shown in Figure 3, was generated using algorithms in both SCIRun [1] and CUBIT [3], and contains 224,496 hexahedra in the engine casing and an additional 274,095 hexahedra in the engine interior. The mesh is completely conformal throughout the model, but is separated into the two material blocks. A transparent view of the geometry showing the engine casing and interior features is shown in Figure 4.

The hexahedral mesh of the engine was generated by creating a simplified geometry that could be meshed using a sweeping algorithm in CUBIT (see Figure 5). A cube was placed around the lower portion of the engine exposing part of the engine neck before the heat fins on the upper neck. A set of new sheets was added to the mesh of this simplified geometry to capture the features of the lower external engine geometry. The hexahedral elements external to the engine were discarded, as shown in the process flow diagram in Figure 5.

The quadrilaterals on the boundary of this mesh were smoothed using a centroidal area smoothing algorithm in CUBIT, and the hexahedra were smoothed using Laplacian smoothing, mesh untangling and condition number optimization. A second set of sheets was then inserted to capture the internal surface of the engine. The new internal boundary was smoothed with

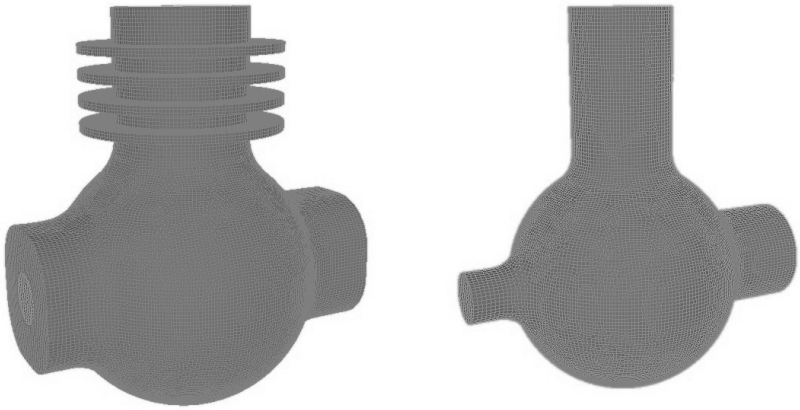

Fig. 3. Hexahedral mesh of the engine model. The interior hexahedral mesh for the engine model is shown on the right.

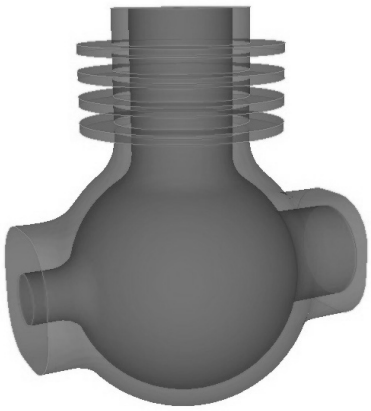

Fig. 4. Transparent view of the engine and interior created from the facets of the hexahedral mesh of the engine model.

centroidal area smoothing, and the hexahedra were again smoothed with a Laplacian smoothing algorithm. Both meshes were then optimized against the condition number metric. The distribution of element quality for the scaled Jacobian metric for the engine mesh is shown in Figure 6.

3.2 Part29 Model

The Part29 model is not readily sweepable, nor does it contain any recognizably dominant hexahedral primitive shapes making traditional hexahedral

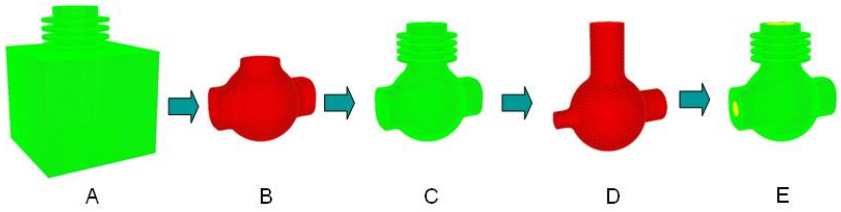

Fig. 5. Pictorial flow chart demonstrating the mesh generation process for creating the hexahedral mesh of the engine. Triangle meshes (red) are utilized to guide placement of hexahedral sheets into the existing hexahedral mesh to achieve a new mesh that is conformal with the original solid geometry.

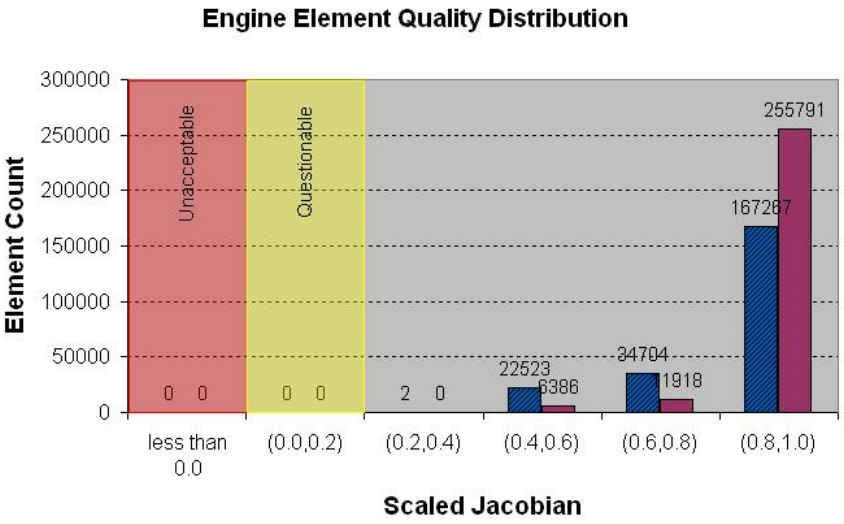

Fig. 6. Distribution of element quality for engine model (hexahedral statistics for the interior of the engine are shown in white, and statistics for the engine casing are shown in black.

meshing with traditional algorithms difficult. The hexahedral mesh, shown in Figure 7 contains 17,386 hexahedra. To generate this mesh, a simplified, multisweepable [6] model was created (as shown in Figure 8) and meshed in CUBIT. A set of sheets, similar in shape to the upper surface, was inserted into the mesh, and the elements above this sheet were discarded. Then, a cylindrical hole was added by inserting an additional set of sheets using a triangle mesh on a cylindrical surface to guide the placement of these sheets.

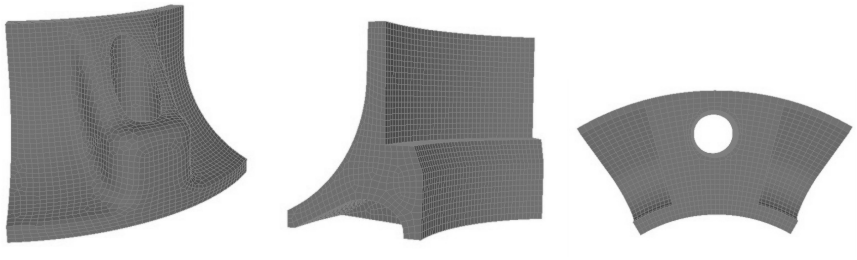

Fig. 7. Hexahedral mesh of the part29 model. Images of the mesh from the front, back and bottom are shown, respectively.

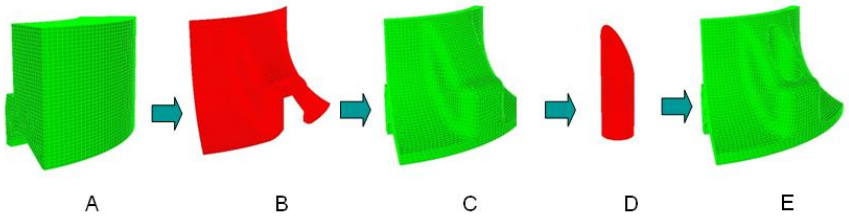

Fig. 8. Pictorial flow chart demonstrating the mesh generation process for creating the hexahedral mesh of part29. Triangle meshes (red) are utilized to guide placement of hexahedral sheets into the existing hexahedral mesh to achieve a new mesh that is conformal with the original solid geometry.

The resulting mesh was smoothed using a centroidal area smoothing algorithm on the surfaces, and a Laplacian smoothing algorithm on the hexahedral elements. Finally, the entire mesh was optimized using a mean-ratio metric to obtain the mesh quality distribution shown in Figure 9.

The sheet insertion process in this example creates highly skewed elements in the front filleted area due to the near tangency of the two sheets used to capture the geometric feature. The skewed elements are the 27 questionable elements shown in the mesh quality distribution in Figure 9. It is possible to improve the quality of these elements using pillowing, as described by Mitchell, et al. [4].

4 Conclusion

The process of geometric decomposition typically used when generating hexahedral meshes can be extremely difficult to perform and automate. Using direct sheet-based techniques, we can generate hexahedral meshes in complex parts with significantly less geometric decomposition than traditional

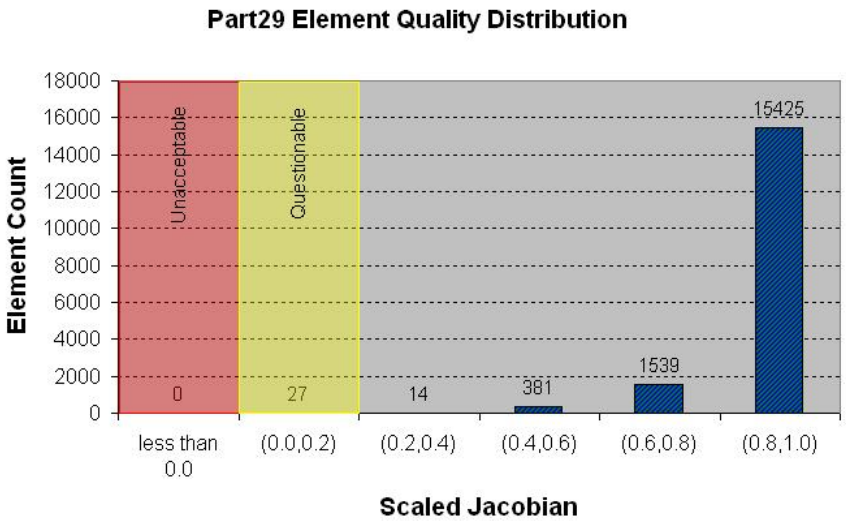

Fig. 9. Distribution of element quality for the part29 model.

hexahedral meshing methods. Sheet insertion is relatively straight-forward to develop, and results in reasonable element quality. Applying multiple sheet insertions, it is possible to capture geometric curves in the resulting mesh topology while maintaining reasonable element quality in the curve's vicinity. This technique is also suitable for building complex hexahedral meshes which have been historically difficult to obtain.

References

1. 2007. SCIRun: A Scientific Computing Problem Solving Environment, Scientific Computing and Imaging Institute (SCI), Download from: http://software.sci.utah.edu/scirun.html.
2. ANSYS. ANSYS, http://www.ansys.com, January 2007.
3. The CUBIT Geometry and Mesh Generation Toolkit, Sandia National Laboratories, http://cubit.sandia.gov/, 2007.
4. S. A. Mitchell and T. J. Tautges. Pillowing doublets: Refining a mesh to ensure that faces share at most one edge. In *Proceedings, 4th International Meshing Roundtable*, pages 231–240. Sandia National Laboratories, October 1995.
5. J. F. Shepherd. *Topologic and Geometric Constraint-Based Hexahedral Mesh Generation.* Published Doctoral Dissertation, University of Utah, May 2007.
6. J. F. Shepherd, S. A. Mitchell, P. Knupp, and D. R. White. Methods for multi-sweep automation. In *Proceedings, 9th International Meshing Roundtable*, pages 77–87. Sandia National Laboratories, October 2000.